Image Segmentation: Otsu’s method of thresholding

Kevin Kemmerer

Image Thresholding vs Image Segmentation

“Image segmentation refers to the class of algorithms that partition the image into different segments or groups of pixels.”

image thresholding is the simplest kind of image segmentation because it partitions the image into two groups of pixels — white for foreground, and black for background

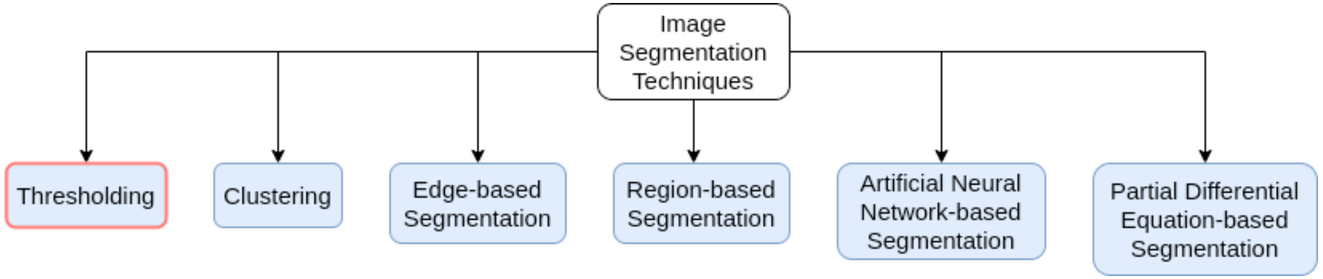
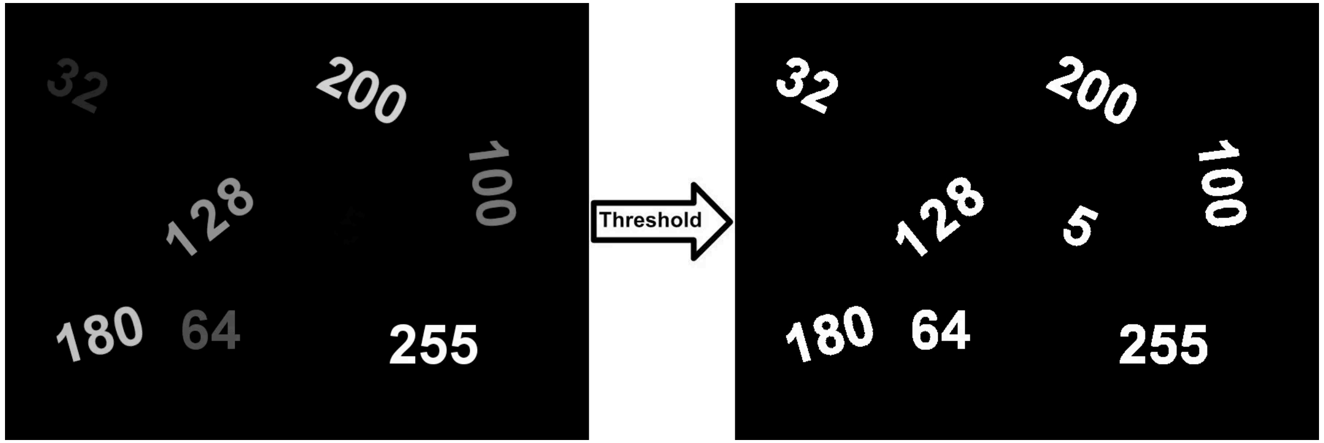


Image Segmentation Techniques

- Thresholding
- Clustering
- Edge-based Segmentation
- Region-based Segmentation
- Artificial Neural Network-based Segmentation
- Partial Differential Equation-based Segmentation

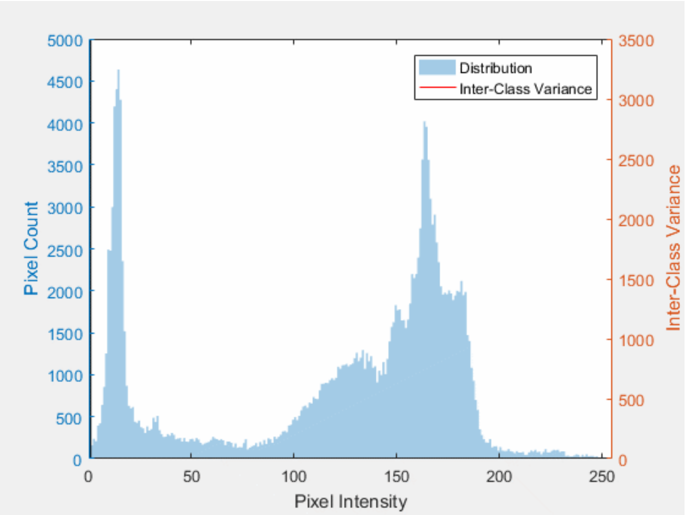


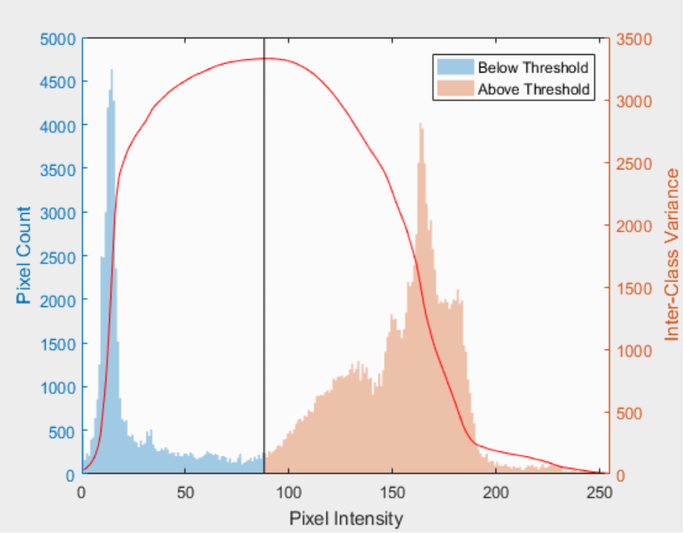
What is Otsu’s method of thresholding?

Thresholding algorithm steps:

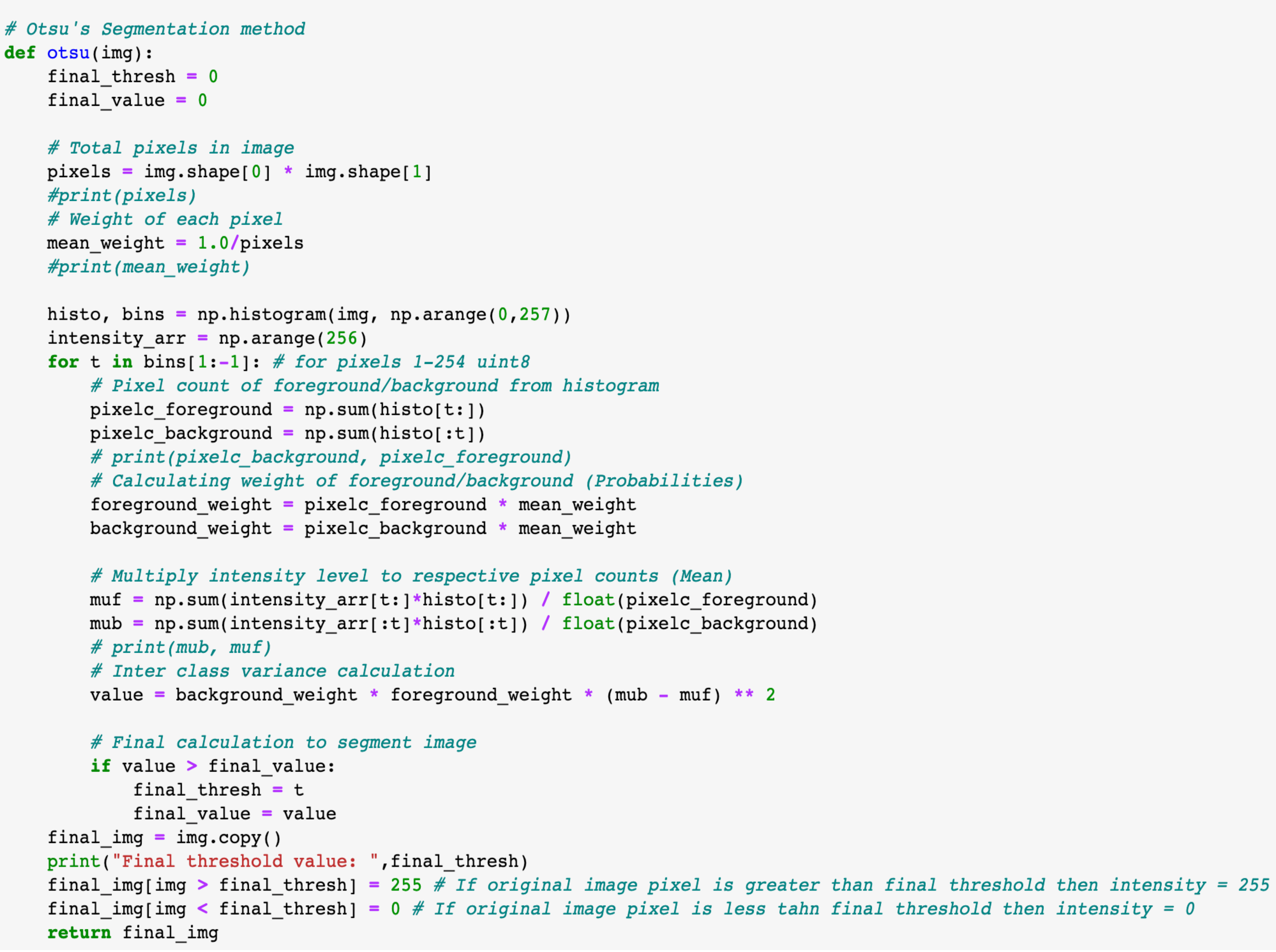
1.Process the input image
2.Obtain image histogram (distribution of pixels)
3.Compute the threshold value *T*
4.Change image pixels to white in those regions, where intensity is greater than *T* and into black where intensity is less than *T*.

3. Computing the threshold value - The method processes image histograms, segmenting the objects by minimization or maximization of variance on each of the classes. The histogram of such image contains two clearly expressed peaks, which represent different ranges of intensity values





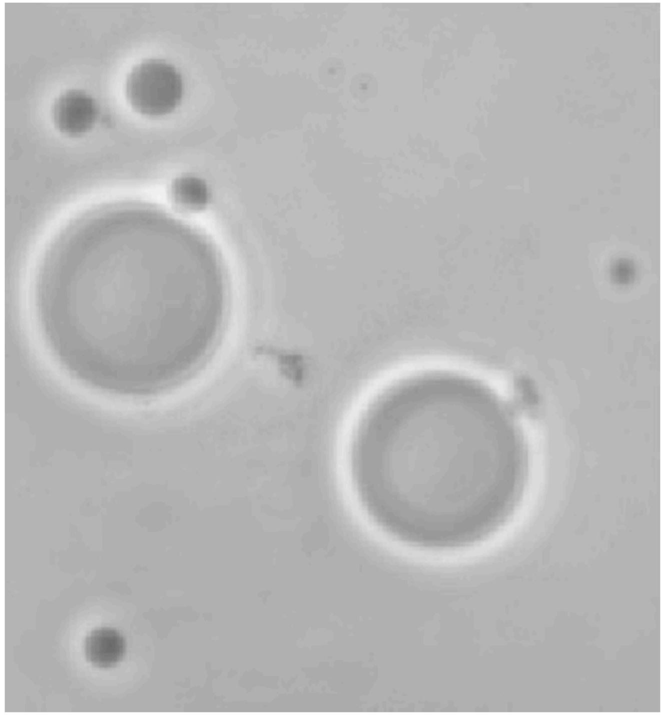
1.Get total pixels/weight
2.Obtain image histogram (distribution of pixels)
3.Calculate the histogram and intensity level probabilities
4.Iterate over possible thresholds *T = 0,..., max_intensity*
5.Calculate the maximum inter class variance
6.Change image pixels to white in those regions, where intensity is greater than *T* and into the black in the opposite cases.



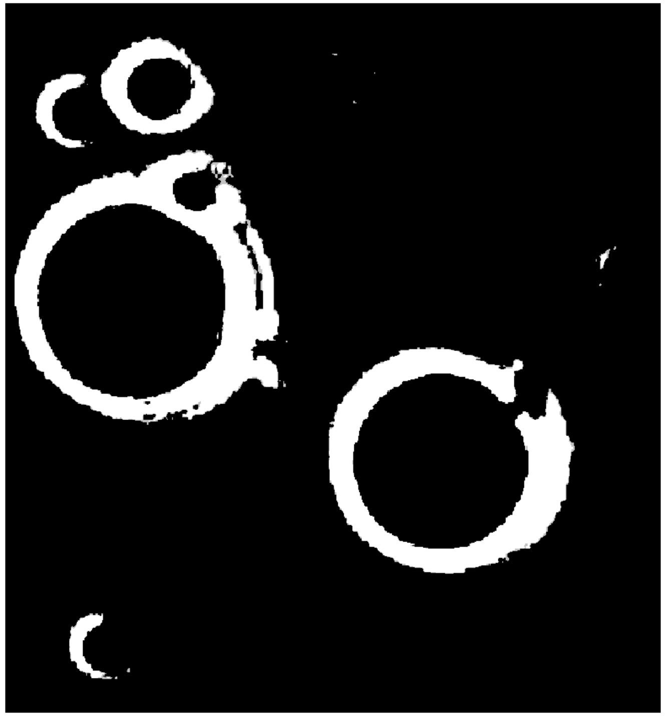
```
# Otsu's Segmentation method
def otsu(img):
    final_thresh = 0
    final_value = 0

    # Total pixels in image
    pixels = img.shape[0] * img.shape[1]
    #print(pixels)
    # Weight of each pixel
    mean_weight = 1.0/pixels
    #print(mean_weight)

    histo, bins = np.histogram(img, np.arange(0,257))
    intensity_arr = np.arange(256)
    for t in bins[1:-1]: # for pixels 1-254 uint8
        # Pixel count of foreground/background from histogram
        pixelc_foreground = np.sum(histo[t:])
        pixelc_background = np.sum(histo[:t])
        # print(pixelc_background, pixelc_foreground)
        # Calculating weight of foreground/background (Probabilities)
        foreground_weight = pixelc_foreground * mean_weight
        background_weight = pixelc_background * mean_weight

        # Multiply intensity level to respective pixel counts (Mean)
        muf = np.sum(intensity_arr[t:]*histo[t:]) / float(pixelc_foreground)
        mub = np.sum(intensity_arr[:t]*histo[:t]) / float(pixelc_background)
        # print(mub, muf)
        # Inter class variance calculation
        value = background_weight * foreground_weight * (mub - muf) ** 2

        # Final calculation to segment image
        if value > final_value:
            final_thresh = t
            final_value = value
    final_img = img.copy()
    print("Final threshold value: ",final_thresh)
    final_img[img > final_thresh] = 255 # If original image pixel is greater than final threshold then intensity = 255
    final_img[img < final_thresh] = 0 # If original image pixel is less tahn final threshold then intensity = 0
    return final_img
```

$\sigma_b^2(t) = w_1(t)w_2(t)[\mu_1(t) - \mu_2(t)]^2$, where μ_i is a mean of class i .

Original Image



Otsu's thresholding result

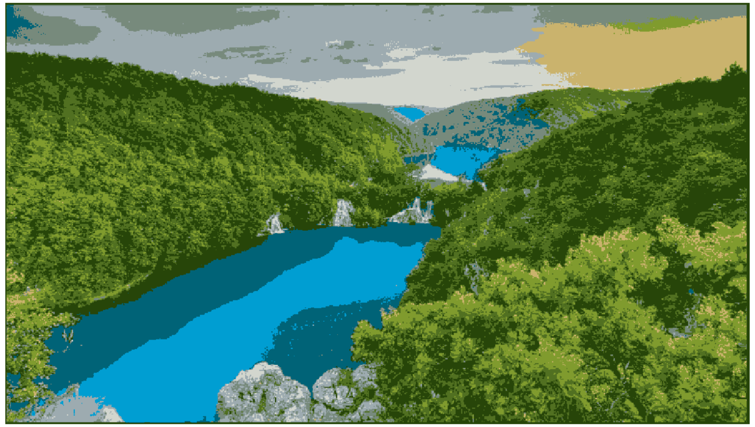


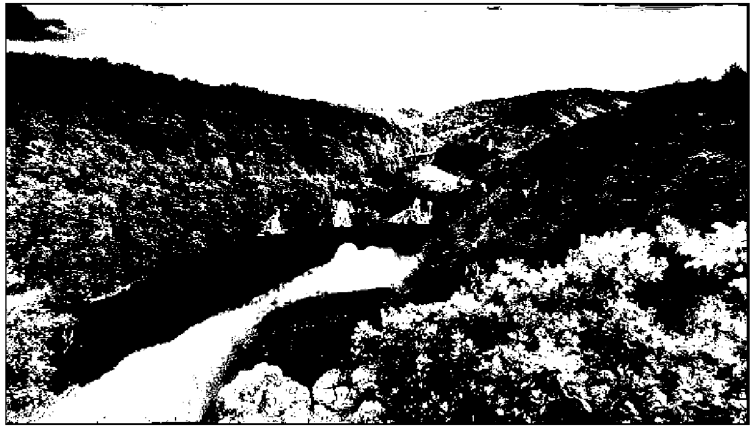
Final threshold value: 182

Otsu's thresholding vs K-Means Clustering

Clustering is the process of using machine learning and algorithms to identify how different types of data are related and creating new segments based on those relationships (Value of K determines how many clusters)

Thresholding is the simplest form of segmentation that usually begins with a grey image and results in only two intensities. Black for background and white for foreground.





Sources

https://en.wikipedia.org/wiki/Otsu%27s_method (For histogram example)

https://learnopencv.com/otsu-thresholding-with-opencv/